IN THE CLAIMS:

Please amend the claims such that the pending claims read as follows:

15. (Thrice Amended) A method, including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in a cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes steps of recording said network objects in said cache memory and retrieving said network objects from said cache memory, so as to substantially minimizes a time required for retrieving said network objects from said mass storage.

16. (Amended) A method as in claim 15, wherein said network objects include an HTML page to be retrieved from said cache memory and served to the client for display.

17. (Amended) A method as in claim 15, including a step of serving said network objects to said client in place of said server.

18. (Amended) A method as in claim 17, wherein said network objects are served to said client in place of said server in response to a second request from said client.

19. A method as in claim 15, wherein said step of receiving uses a computer network.

20. A method as in claim 15, wherein said step of receiving is responsive to protocol messages using a computer network, said protocol messages including a resource identifier for each said network object.

21. A method as in claim 17, wherein said step of serving is responsive to a resource identifier associated with each said network object.

22. A method as in claim 17, wherein said step of serving is responsive to a uniform resource locator associated with each said network object.

39. (Thrice Amended) A method, including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in a cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes steps of recording said network objects in said cache memory and retrieving said network objects from said cache memory, so as to perform at least one of:

maximizing a rate at which said network objects can be written to said mass storage,

maximizing a rate at which said network objects can be erased from said mass storage,

maximizing a rate at which said network objects can be retrieved from said mass storage, or

minimizing a time required for retrieving said network objects from said mass storage.

40. (Amended) A method as in claim 39, wherein said network objects include an HTML page to be retrieved from said cache memory and served to the client for display.

41. (Amended) A method as in claim 39, including a step of serving said network objects to said client in place of said server.

42. (Amended) A method as in claim 41, wherein said network objects are served to said client in place of said server in response to a second request from said client.

43. A method as in claim 39, wherein said step of receiving uses a computer network.

44. A method as in claim 39, wherein said step of receiving is responsive to protocol messages using a computer network, said protocol messages including a resource identifier for each said network object.

45. A method as in claim 41, wherein said step of serving is responsive to a resource identifier associated with each said network object.

46. A method as in claim 41, wherein said step of serving is responsive to a uniform resource locator associated with each said network object.

47. (Twice Amended) A method, including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in a cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining is performed independently of a file system for said mass storage.

54. (Twice Amended) A method, including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in a cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes a step of writing a group of network objects to said mass storage in one or more write episodes, such that said write episodes are performed so as to atomically commit changes to said mass storage during each said write episode by writing modified data and control blocks to the mass storage without erasing corresponding unmodified data and control blocks and then replacing a root node so as to atomically commit the changes.

55. (Twice Amended) A method, including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in a cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes a step of deleting a group of network objects to said mass storage in one or more delete episodes, such that said delete episodes are performed so as to atomically commit changes to said mass storage during each said delete episode by writing modified control blocks to the mass storage without erasing corresponding unmodified control blocks and then replacing a root node so as to atomically commit the changes.

56. (New) A cache engine, including:

a cache memory;

an interface for interfacing to a network;

a processor; and

a memory storing information including instructions, the instructions executable by said processor, the instructions including steps of: (a) receiving a set of network objects in response to a first request to a server from a client; and (b) maintaining said network objects in said cache memory in said cache engine, said cache engine connected via the network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes steps of recording said network objects in said cache memory and retrieving said network objects from said cache memory, so as to substantially minimizes a time required for retrieving said network objects from said mass storage.

57. (New) A cache engine as in claim 56, wherein said network objects include an HTML page to be retrieved from said cache memory and served to the client for display.

58. (New) A cache engine as in claim 56, wherein the instructions further include a step of serving said network objects to said client in place of said server.

59. (New) A cache engine as in claim 58, wherein said network objects are served to said client in place of said server in response to a second request from said client.

60. (New) A cache engine as in claim 56, wherein said step of receiving uses a computer network.

61. (New) A cache engine as in claim 56, wherein said step of receiving is responsive to protocol messages using a computer network, said protocol messages including a resource identifier for each said network object.

62. (New) A cache engine as in claim 58, wherein said step of serving is responsive to a resource identifier associated with each said network object.

63. (New) A cache engine as in claim 58, wherein said step of serving is responsive to a uniform resource locator associated with each said network object.

64. (New) A cache engine, including:

a cache memory;

an interface for interfacing to a network;

a processor; and

a memory storing information including instructions, the instructions executable by said processor, the instructions including steps of: (a) receiving a set of network objects in response to a first request to a server from a client; and (b) maintaining said network objects in said cache memory in said cache engine, said cache engine connected via the network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes steps of recording said network objects in said cache memory and retrieving said network objects from said cache memory, so as to perform at least one of:

maximizing a rate at which said network objects can be written to said mass storage, maximizing a rate at which said network objects can be erased from said mass storage,

maximizing a rate at which said network objects can be retrieved from said mass storage, or

minimizing a time required for retrieving said network objects from said mass storage.

65. (New) A cache engine as in claim 64, wherein said network objects include an HTML page to be retrieved from said cache memory and served to the client for display.

66. (New) A cache engine as in claim 64, wherein the instructions further include a step of serving said network objects to said client in place of said server.

67. (New) A cache engine as in claim 66, wherein said network objects are served to said client in place of said server in response to a second request from said client.

68. (New) A cache engine as in claim 64, wherein said step of receiving uses a computer network.

69. (New) A cache engine as in claim 64, wherein said step of receiving is responsive to protocol messages using a computer network, said protocol messages including a resource identifier for each said network object.

70. (New) A cache engine as in claim 66, wherein said step of serving is responsive to a resource identifier associated with each said network object.

71. (New) A cache engine as in claim 66, wherein said step of serving is responsive to a uniform resource locator associated with each said network object.

72. (New) A cache engine, including:

a cache memory;

an interface for interfacing to a network;

a processor; and

a memory storing information including instructions, the instructions executable by said processor, the instructions including steps of: (a) receiving a set of network objects in response to a first request to a server from a client; and (b) maintaining said network objects in said cache memory in said cache engine, said cache engine connected via the network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining is performed independently of a file system for said mass storage.

73. (New) A cache engine, including:

a cache memory;

an interface for interfacing to a network;

a processor; and

a memory storing information including instructions, the instructions executable by said processor, the instructions including steps of: (a) receiving a set of network objects in response to a first request to a server from a client; and (b) maintaining said network objects in said cache memory in said cache engine, said cache engine connected via the network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes a step of writing a group of network objects to said mass storage in one or more write episodes, such that said write episodes are performed so as to atomically commit changes to said mass storage during each said write episode by writing modified data and control blocks to the mass storage without erasing corresponding unmodified data and control blocks and then replacing a root node so as to atomically commit the changes.

74. (New) A cache engine, including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in a cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes a step of deleting a group of network objects to said mass storage in one or more delete episodes, such that said delete episodes are performed so as to atomically commit changes to said mass storage during each said delete episode by writing modified control blocks to the mass storage without erasing corresponding unmodified control blocks and then replacing a root node so as to atomically commit the changes.

75. (New) A memory storing information including instructions, the instructions executable by a processor to control a cache engine, the instructions including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in the cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes steps of recording said network objects in said cache memory and retrieving said network objects from said cache memory, so as to substantially minimizes a time required for retrieving said network objects from said mass storage.

76. (New) A memory as in claim 75, wherein said network objects include an HTML page to be retrieved from said cache memory and served to the client for display.

77. (New) A memory as in claim 75, wherein the instructions further include a step of serving said network objects to said client in place of said server.

78. (New) A memory as in claim 77, wherein said network objects are served to said client in place of said server in response to a second request from said client.

79. (New) A memory as in claim 75, wherein said step of receiving uses a computer network.

80. (New) A memory as in claim 75, wherein said step of receiving is responsive to protocol messages using a computer network, said protocol messages including a resource identifier for each said network object.

81. (New) A memory as in claim 77, wherein said step of serving is responsive to a resource identifier associated with each said network object.

82. (New) A memory as in claim 77, wherein said step of serving is responsive to a uniform resource locator associated with each said network object.

83. (New) A memory storing information including instructions, the instructions executable by a processor to control a cache engine, the instructions including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in the cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes steps of recording said network objects in said cache memory and retrieving said network objects from said cache memory, so as to perform at least one of:

maximizing a rate at which said network objects can be written to said mass storage,

maximizing a rate at which said network objects can be erased from said mass storage,

maximizing a rate at which said network objects can be retrieved from said mass storage, or

minimizing a time required for retrieving said network objects from said mass storage.

84. (New) A memory as in claim 83, wherein said network objects include an HTML page to be retrieved from said cache memory and served to the client for display.

85. (New) A memory as in claim 83, wherein the instructions further include a step of serving said network objects to said client in place of said server.

86. (New) A memory as in claim 85, wherein said network objects are served to said client in place of said server in response to a second request from said client.

87. (New) A memory as in claim 83, wherein said step of receiving uses a computer network.

88. (New) A memory as in claim 83, wherein said step of receiving is responsive to protocol messages using a computer network, said protocol messages including a resource identifier for each said network object.

89. (New) A memory as in claim 85, wherein said step of serving is responsive to a resource identifier associated with each said network object.

90. (New) A memory as in claim 85, wherein said step of serving is responsive to a uniform resource locator associated with each said network object.

91. (New) A memory storing information including instructions, the instructions executable by a processor to control a cache engine, the instructions including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in the cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining is performed independently of a file system for said mass storage.

92. (New) A memory storing information including instructions, the instructions executable by a processor to control a cache engine, the instructions including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in the cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes a step of writing a group of network objects to said mass storage in one or more write episodes, such that said write episodes are performed so as to atomically commit changes to said mass storage during each said write episode by writing modified data and control blocks to the mass storage without erasing corresponding unmodified data and control blocks and then replacing a root node so as to atomically commit the changes.

93. (New) A memory storing information including instructions, the instructions executable by a processor to control a cache engine, the instructions including steps of:

receiving a set of network objects in response to a first request to a server from a client; and

maintaining said network objects in a cache memory in the cache engine, said cache engine connected via a network to the server and the client, said cache memory including mass storage;

wherein said step of maintaining includes a step of deleting a group of network objects to said mass storage in one or more delete episodes, such that said delete episodes are performed so as to atomically commit changes to said mass storage during each said delete episode by writing modified control blocks to the mass storage without erasing corresponding unmodified control blocks and then replacing a root node so as to atomically commit the changes.